REMARKS

Claims 1-19 are currently pending. Claims 3, 8 and 15 have been amended herein. The amendments to the claims do not present new matter. Reconsideration of the rejection of the pending claims is respectfully requested in view of the above amendments and the following remarks.

Claims 3, 8, 9, 14, and 18 have been rejected under 35 U.S.C. § 112, second paragraph, as being indefinite due to informalities and/or lack of antecedent basis for the element "the plurality of threads." Claim 3 has been amended to correct the noted informality, and claim 8 has been amended to correct the lack of antecedent basis for the above-recited element. As claims 9, 14 and 18 depend from claim 8, proper antecedent basis has been provided by the amendment to claim 8. It is respectfully submitted that claims 3, 8, 9, 14, and 18 are definite. Withdrawal of the indefiniteness rejection is therefore respectfully requested.

Claim 15 has been rejected under 35 U.S.C. § 101 as being directed to nonstatutory subject matter. Specifically, the Office Action asserts that claim 15, as an apparatus claim, must reference something tangible such as software. While it is respectfully submitted that those of skill in the art would recognize that the recited elements – the call flow engine, call flow manager and call flow event queues – represent functional elements of a processor or configurable processing element, in order to facilitate expedited prosecution, claim 15 has been amended to recite a processor that includes the further recited elements of the claim. Since it is well accepted in the art that a processor constitutes a tangible hardware electronic device, it is submitted that claim 15 is directed to statutory subject matter. Withdrawal of the rejection of claim 15 under 35 U.S.C. § 101 is therefore respectfully requested.

Claims 1-5, 7-12 and 14-18 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Sudo (U.S. Patent No. 5,692,192) over Chang et al. (U.S. Patent No. 6,338,078) ("Chang").

Independent claim 1 recites a method, performed at a manager, of distributing call flow events among a plurality of threads, each thread having an associated call

flow event queue in which call flow events are queued, the method comprising, *inter alia*, determining a call flow workload level for each of the plurality of threads.

The Sudo reference concerns a system and method for load balancing among nodes (processing units) of a network whereby distributed tasks (each of which include one or more threads) are either expanded or compressed among the various nodes so as to spread the tasks equitably among the nodes. See Sudo, col. 7, lines 7-59. It is noted that in this method, Sudo monitors the load on nodes, that is to say, the load on the various processing units in network. There is no indication that Sudo monitors or determines a workload level for the threads of which the various tasks are comprised. In fact, in Sudo, the threads are taken to be irreducible entities that are manipulated, as whole units, by being moved from one node to another. Thus, it is submitted that Sudo fails to disclose or suggest the step of determining a call flow workload level at the thread level. Since Sudo does not make a determination as to the load state at the thread level, Sudo necessarily fails to disclose or suggest the further steps of determining that a first of the plurality of threads is inefficiently handling its assigned call flow workload or reassigning a call flow event from the call flow event queue associated with the first thread to the call flow event queue associated with a second of the plurality of threads.

The Chang reference concerns a multiprocessor system which increases the parallelism (and the hence the efficiency) of processing by distributing IP queues among the CPUs in the multiprocessor system. Specifically, Chang prescribes allocating every inbound packet onto an IP queue "wherein the number of IP queues will equal the number of CPUs in the system." Chang, col. 5, lines 16-17. In this method, an IP queue is automatically split among a number of threads, "with one thread per CPU, and one queue per thread." Chang, col. 5, lines 21-22.

Even if it may be argued that Chang prescribes allocation of queued information among a plurality of threads, this allocation occurs according to an automatic funneling of queued information among the various CPUs and does not involve the monitoring of the threads during their operation on the various respective CPUs. Therefore, Chang also does not disclose or suggest the step of determining a call flow workload level at the thread level. In fact, according to its automatic procedure, Chang does not refer to or suggest making any determinations as to

workload prior to allocation.

In sum, it is submitted that the present invention teaches (and recites) a software-level load balancing method whereby a plurality of threads (which may be running on a single processor) are monitored. If there is an imbalance among the threads, individual queued events can be transferred from a heavily-loaded thread to a more lightly-loaded thread. While Sudo refers to a load balancing scheme, it concerns monitoring of processors and allocating whole threads among processors. It is thus a processor-level load balancing method, rather than a software-level load balancing scheme. These schemes differ in that software-level load balancing seeks to remove bottlenecks within sequential instructional queues, whereas processor-level load balancing seeks to balance allocation of processing resources in general. While the Chang reference refers to instructional queues, any apparent similarity to the present invention is misleading because Chang does not refer to, or concern, a balancing among instructional queues through monitoring and transfer of events; instead, Chang simply divides a single long instructional queue into several smaller queues that can each be processed concurrently on separate processors, increasing the speed at which the queue is processed. Thus, even if a skilled practitioner were to consult the Sudo and Chang references in combination, the practitioner will be motivated to perform a method of splitting a queue among a plurality of processors and then monitoring the load among the processors, but the skilled practitioner would not be directed or motivated to monitor the loads on a plurality of threads to achieve a software-level balancing among a plurality of threads as provided by the present invention.

It is therefore respectfully submitted that the combination of Sudo and Chang does not disclose or suggest each of the features of claim 1, which is therefore patentable over the applied references. Since claims 2-5, 7 and 17 depend from claim 1, they are likewise patentable over the applied references.

Since independent claim 8 recites a computer program product for use with a computer system including program code configured to determine a call flow workload level for each of a plurality of threads, it is respectfully submitted that claim 7 is also patentable over the applied references for the same reasons given above with respect to claim 1. Since claims 9-12, 14 and 18 depend from claim 8, they are

likewise patentable over the applied references.

Since independent claim 15 recites a processor that includes modules that perform processes analogous to those recited in claim 1, it is submitted that claim 15 and claim 16, which depends from claim 15, are equally patentable over the cited references.

In light of the foregoing, withdrawal of the rejection of claims 1-5, 7-12 and 14-18 under 35 U.S.C. § 103(a) is accordingly respectfully requested.

Claims 6, 13 and 19 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Sudo in view of Chang, and in further view of Boland et al. (U.S. Patent No. 5,872,972).

Claims 6, 13 and 19 depend from, and incorporate the features of, claims 1, 8 and 15, respectively. Since the Boland reference, like the Sudo and Chang references, only refers to affecting load relationship between processors and does not disclose or suggest determining a call flow workload level for each of a plurality of threads, it is submitted that Boland does not cure the deficiencies of the Sudo and Chang references with respect to the independent claims, and thus the combination of the applied references does not disclose each of the features of claims 6, 13 and 19. It is therefore submitted that claims 6, 13 and 19 are patentable over the applied references. Withdrawal of the rejection of claims 6, 13 and 19 is thus respectfully requested.

Conclusion

All issues having been addressed, it is believed that the present application is in condition for allowance. Prompt reconsideration and allowance of the present application are respectfully requested.

Respectfully submitted,

KENYON & KENYON

Date: December 1, 2003

Jong H. Lee
Registration No. 36,197

KENYON & KENYON
One Broadway
New York, NY 10004

CUSTOMER NO. 26646